# Method of Storing Applications on Removable Storage

## TECHNICAL FIELD OF THE INVENTION

[0001] This invention relates to storage management on a computer, and more particularly to storing a computer application program on removable memory or other removable storage.

## BACKGROUND OF THE INVENTION

[0002] Removable storage is becoming widespread and increasingly useful in a variety of computer systems. In most of the new applications, removable memory is used to store data files. It is also advantageous to store applications on removable memory or other removable storage. In typical computer systems, storing applications on the removable memory is cumbersome to install and un-install the programs so that the system maintains a current file extension table without broken links to removed applications on the removable memory.

[0003] The computer system software needs to be able to recognize a new application on the removable memory and have all data types associated with that application to be correctly updated when the memory is attached. The system must also be able to determine when the application is no longer present when the removable memory is removed from the system. This is

necessary to prevent the system from having invalid links to software that has been removed.

[0004] In a classroom environment, the teacher could hand out a removable memory or storage with an application as well as data files for a test or other activity. It is desirable for a computer such as a handheld computer device to be able to efficiently and easily handle the new application and data files into the file system on the device.

## SUMMARY OF THE INVENTION

[0005] The present invention overcomes problems associated with the described prior art. In a preferred embodiment, a separate applications database is maintained on the removable storage that keeps track of data types for the application on the removable storage. Upon insertion the data base on the removable storage can be merged with the applications database on the computer system.

[0006] Preferred embodiments of the present are directed to handheld computer devices such as calculators and personal digital assistants.

[0007] An advantage of an embodiment of the present invention is that applications can be easily integrated with the computers system applications upon insertion of the memory containing the application. In a preferred embodiment the application table is dynamically updated to reflect the applications on any removable memory in such a way that nothing need be stored on the computer to support those applications on the removable memory. Embodiments of the present invention also allow the computer system to keep track of the most up to date copy of the application, whether on the removable memory on the computers main memory.

[0008] Another advantage of the present invention is that the application most likely desired by the user is used to run the desired data type.

## BRIEF DESCRIPTION OF THE DRAWINGS

[0009] FIGURE 1 illustrates a physical table containing an application data base according to the prior art.

[0010] Figure 2 illustrates a computer with a removable memory containing an application program according to an embodiment of the prior art.

[0011] FIGURE 3 illustrates a physical table and a corresponding virtual table containing an application data base according to an embodiment of the present invention.

[0012] FIGURE 4a, b illustrate flow diagrams according to an embodiment of the present invention.

## DETAILED DESCRIPTION OF THE INVENTION

[0013] FIGURE 1 represents a data extension table **10** for an operating system according to the prior art. The table lists a number of data file types that are represented by a file extension **12**. The file extension may be one or more characters and is typically a three letter extension in common computer systems. Each data file type is associated with one or more applications **14** that use that file type. This data table exists as a physical table somewhere in system memory or stored on a system disk file.

[0014] FIGURE 2 illustrates a computer with a removable storage or memory **16** having a data extension table and an application program according to an embodiment of the present invention. The physical data extension tables on the computer and on the removable memory are merged into a virtual table data extension table (not shown). The removable storage **16** is shown inserted into the portable computer **18**.

[0015] FIGURE 3 illustrates a data extension table **100** constructed according to an embodiment of the present invention. This embodiment shows a number of physical tables **102** that are merged into a virtual table **104**. The first physical table is the local table **106**. The local table **106** resides in local memory to the system like in the prior art. There can be one or more additional physical tables **108, 110** that reside on removable memory. In

Figure **2** a second table **108** is shown along with a final table designated as Table n **110**.

[0016] The local physical table **102** includes a data file type **112** represented by a file extension, that is associated with one or more applications **114** that use that file type. Preferably, this table also includes a time stamp **116** for each file extension entry. Similarly the other physical tables **108, 110** include the same entries. The physical tables **102** are merged into a virtual table **104** by the operating system or by a system computer program or routine after detecting the insertion of the removable memory module.

[0017] In a preferred embodiment, the virtual table **104** is constructed to resemble the prior art physical table. The virtual table **104** lists the file types with their associated applications. For data type entries in the virtual table that had more than one application program, the most recent application is listed in the table.

[0018] Figure **3** shows a flow diagram according to an embodiment of the present invention. The flow diagram could be use by the host system to process the physical tables **106** to form the virtual table **104**, or the flow diagram could be used to determine the application to be used without forming the virtual table.

[0019] The flow diagram describes the process of determining which application software to use for a given file type according to the present invention. The invention first gives priority to an application installed after the

insertion of removable memory containing the data file, and second to an application on the most recently installed removable memory and last to an application in physical local memory. Advantageously the present invention associates the application most likely desired by the user to run the desired data type. For example, if a user adds a removable memory which contains both an application and a data type associated with that data type, the user likely wants to use that application. If the user after installing a removable memory installs an application for the data type, the decision process will choose the later installed application since that is likely what the user wants. The system may also allow the user to choose another application for a data file to override this process.

[0020] Figures 4a and 4b show a flow diagram to implement the process described above according to an embodiment of the present invention. The process determines an application to be associated with a file type, usually indicated by the data file extension. The process is shown as a single scan for a given file type, but can be reiterated for each file type to build a virtual data type extension table **104**. The process begins **210** in Figure 4a by looking at the most recently installed application **212** and examining the time stamp in the physical tables **106** in Figure **3**. If the application was installed since the last connection of the module **214**, and the application handles the data type being determined **216** then the routine returns a found application **218** for the current data type. If the application does not handle the data type

and there are more applications to check **220**, then the next newest application **222** is checked. If there are no other applications to check, then not found is returned. This sequence will check for an application installed after the connection to the removable memory containing the data file currently of interest.

[0021] If no applications are found since the last insertion of the module containing the data file, then applications installed before are examined. The applications installed before are examined in order of the most recently connected location **224**. The first most recently connected location is determined from the time stamp associated with the inserted module. The process then proceeds to the Location Search loop **226** shown in Figure **4b**. The location search loop looks at the most recently installed application **228** on the currently being searched location. If the first application handles the needed data type **230**, then it returns a found application to the system. If not, then a check is made for more applications **232**. The next newest application is checked **234** as long as there are more application on the presently being searched location.

[0022] After all applications on the presently being search location are searched, the Location search loop returns a found or not found to the Location search call **226** in the previous loop. The process loop **236** after this call then checks if the location search returned a found application **238**. If the location search returned not found, then the process checks for additional

locations **240** and continues the loop for any additional locations, or returns not found if there are no more locations to search.

Other Embodiments

[0023] Although the present invention has been described in detail, it should be understood that various changes, substitutions, and alterations could be made hereto without departing from the spirit and scope of the invention as defined by the appended claims.